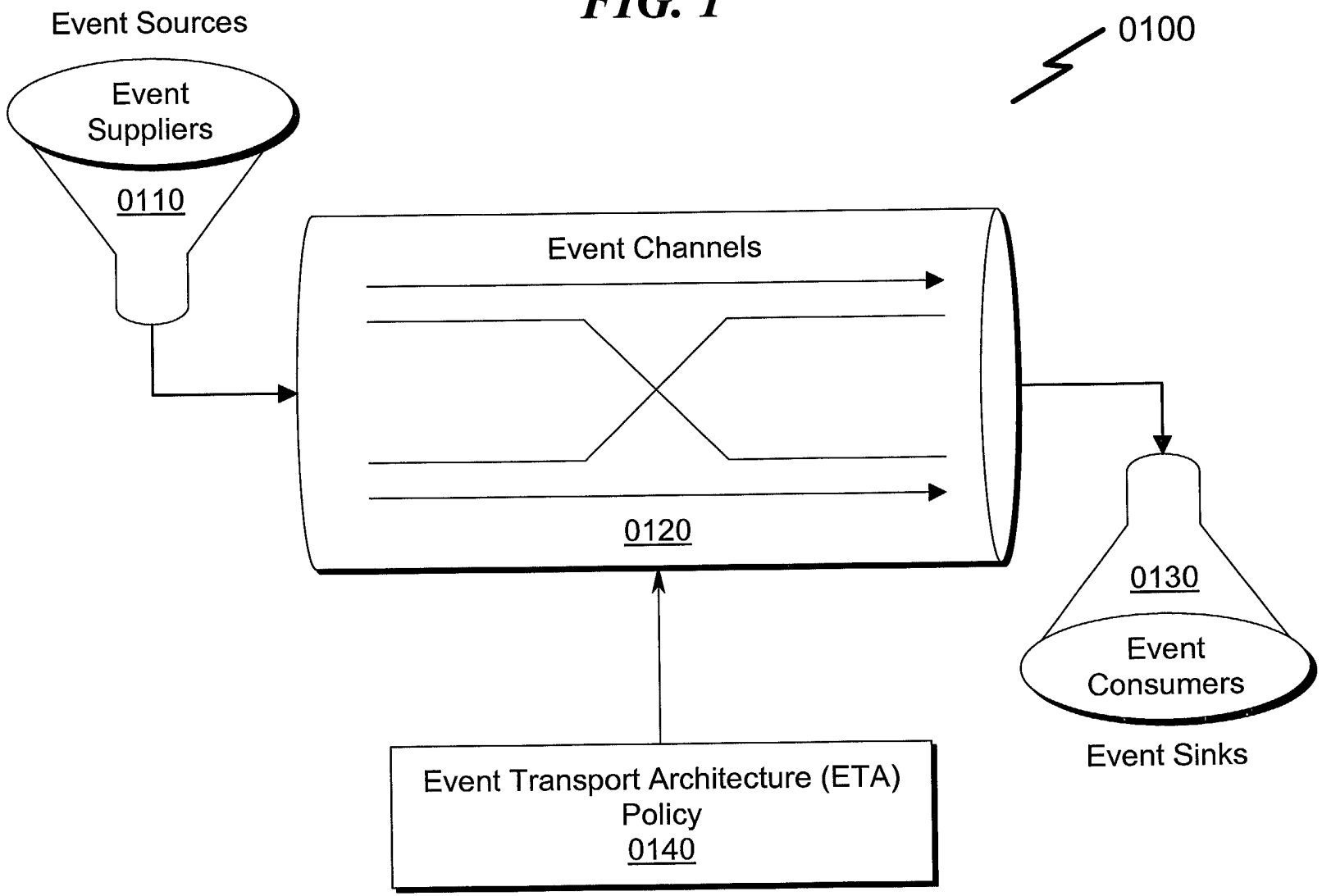
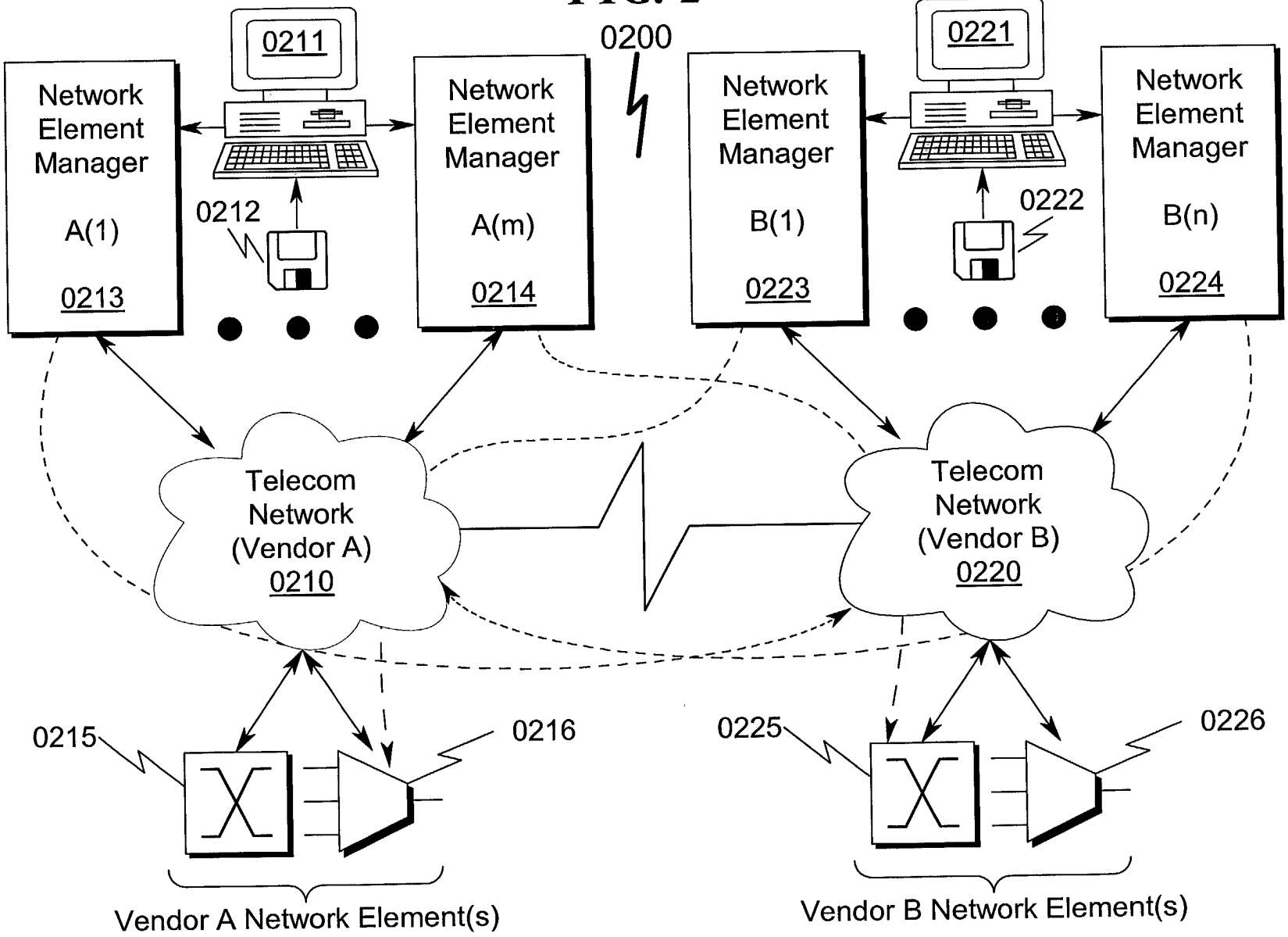
TOTTSO" 9E7ES860

FIG. 1



TOTTSO' 9E/ESS60

FIG. 2



TOTTSO" 9E/ES960

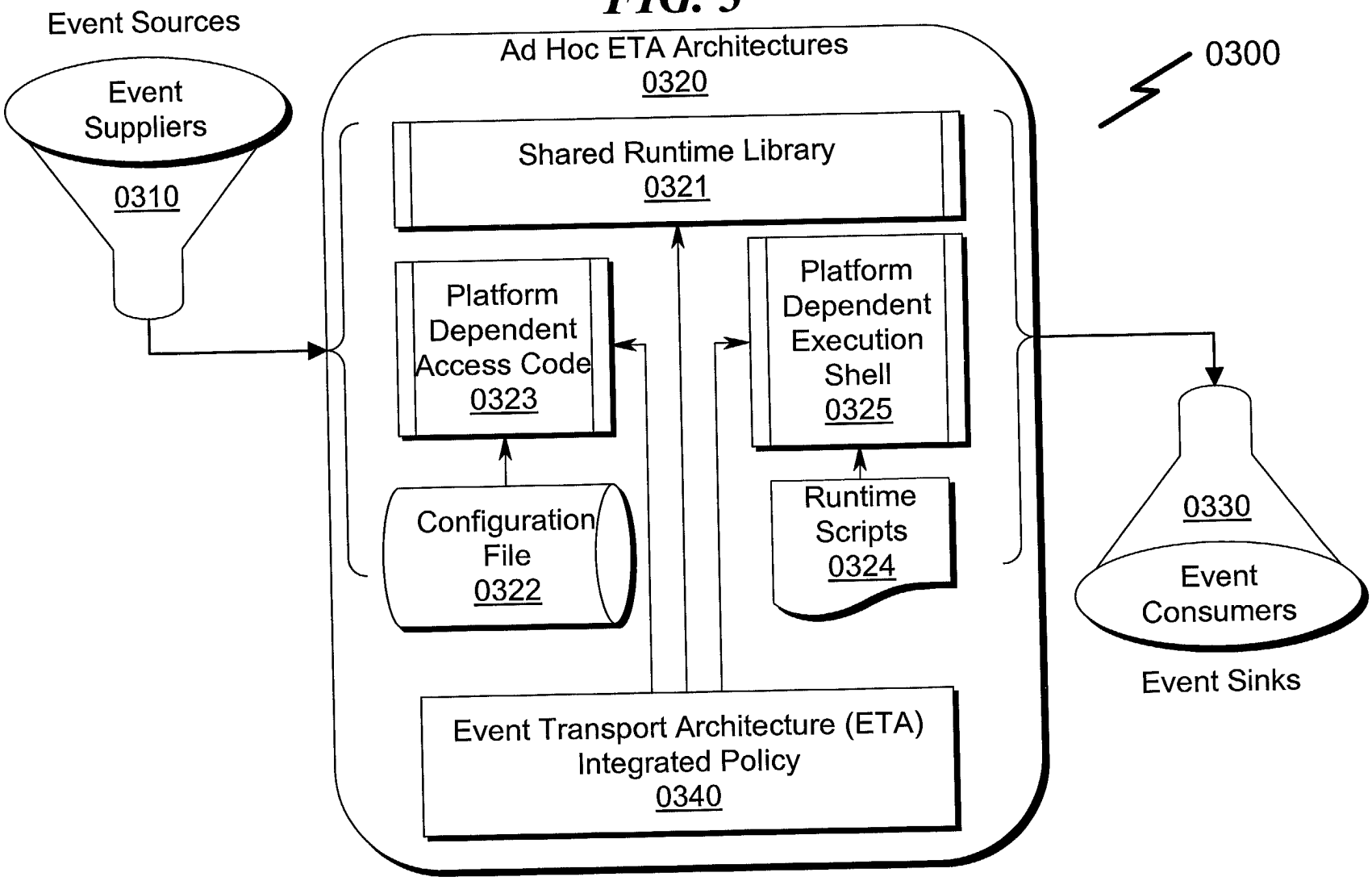
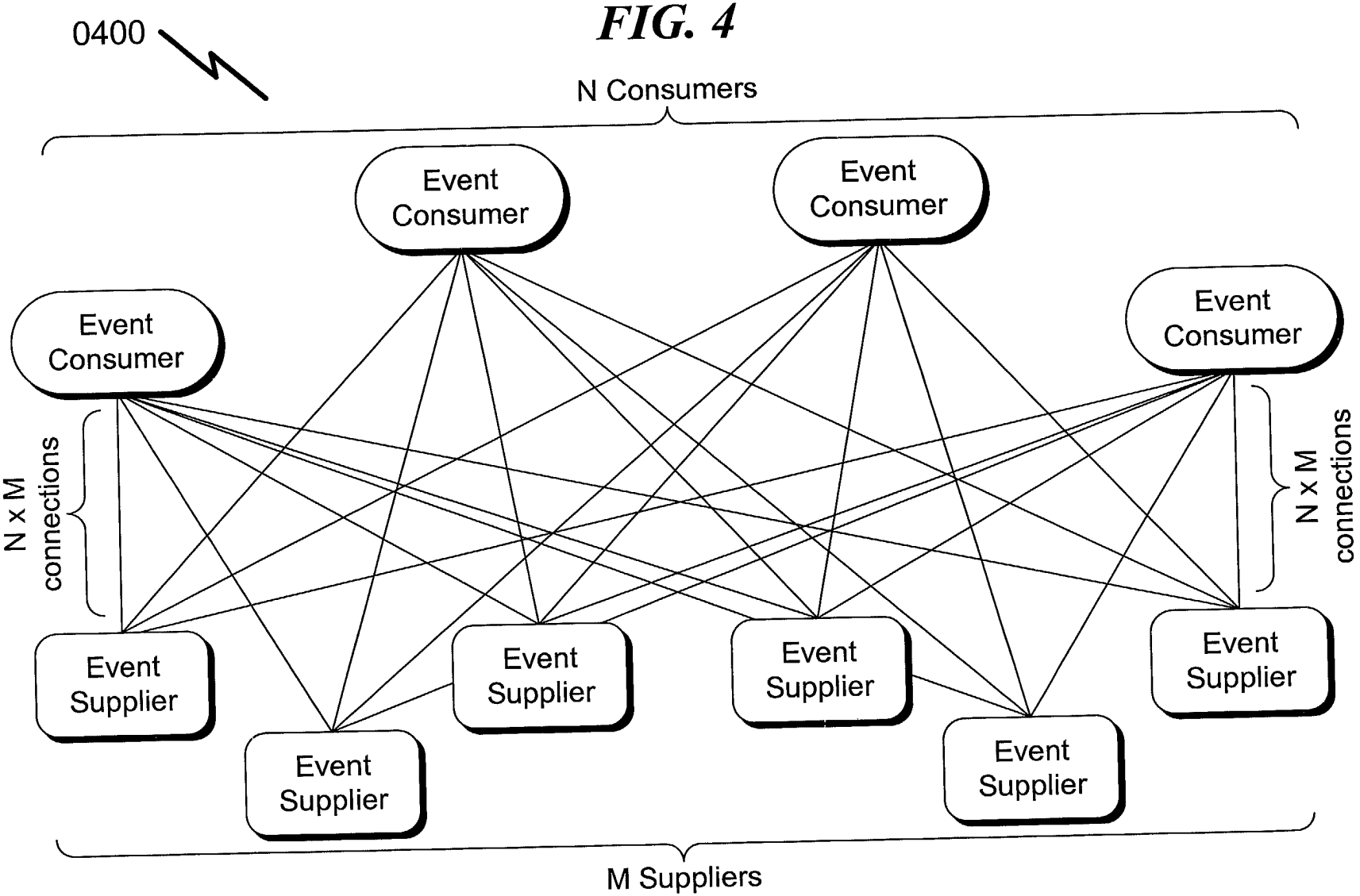


FIG. 3

PRIOR ART

TOTTSO" 9E7E5860

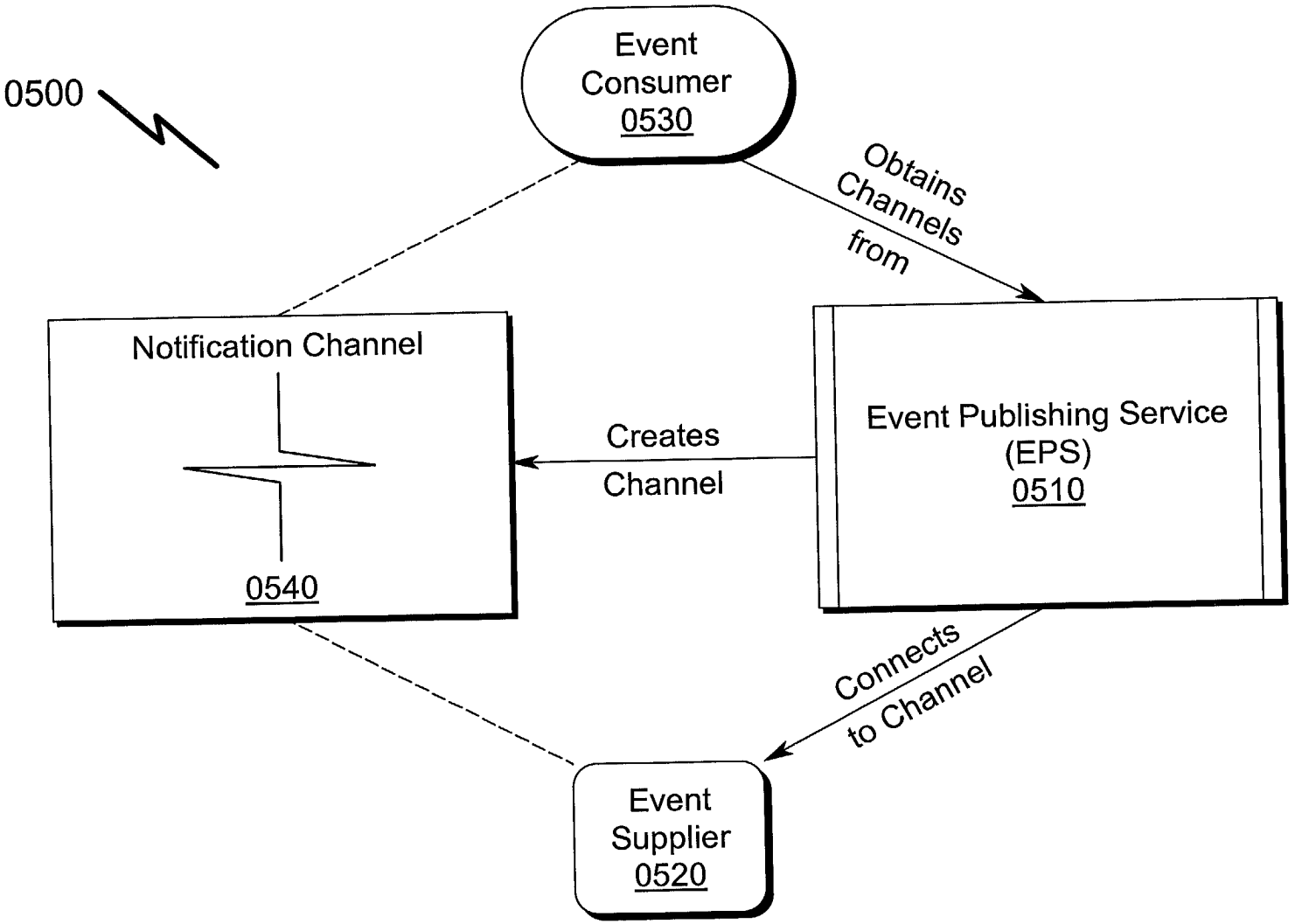
FIG. 4



PRIOR ART

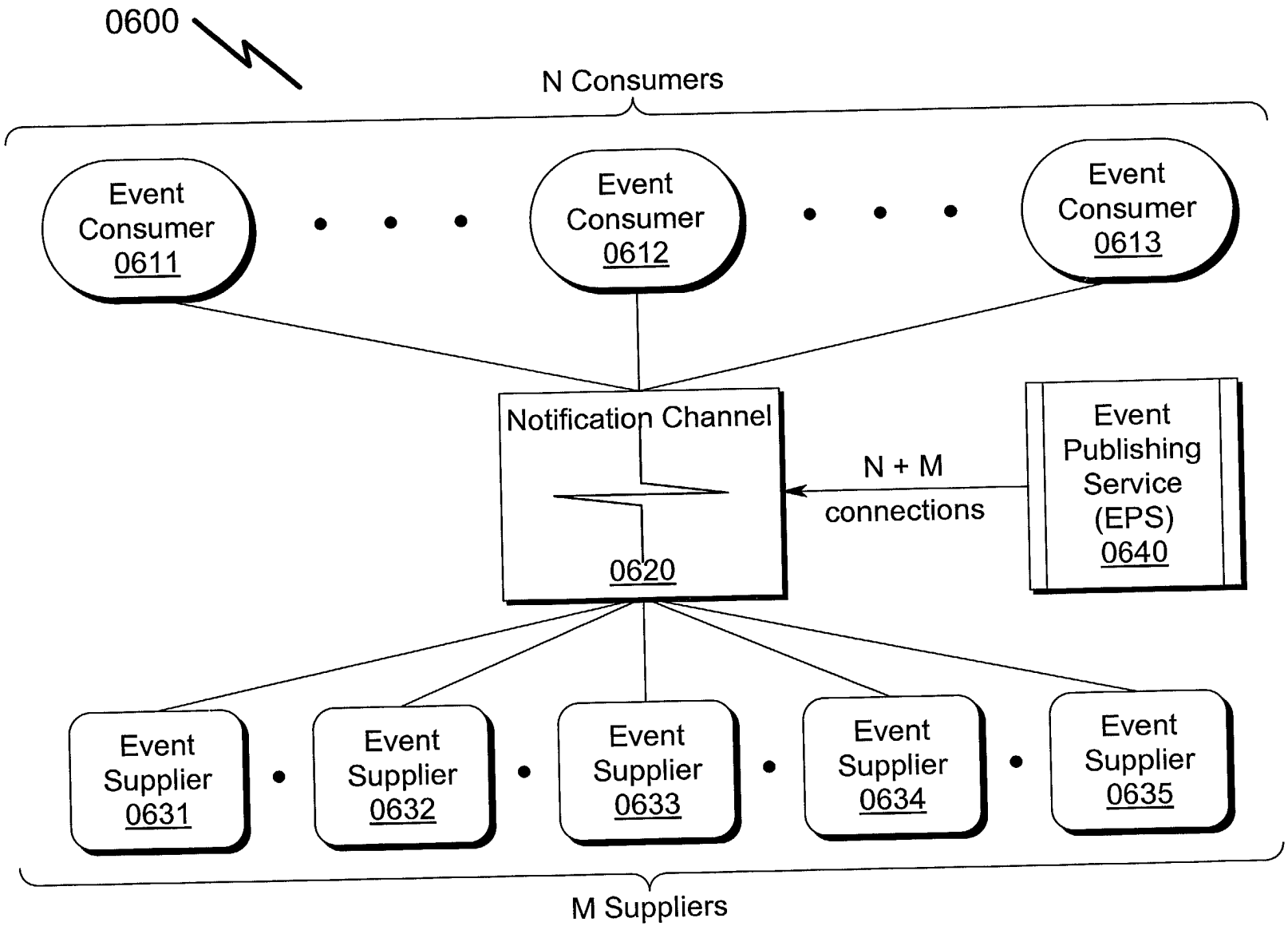
TOTTSO' 9E/ES860

FIG. 5



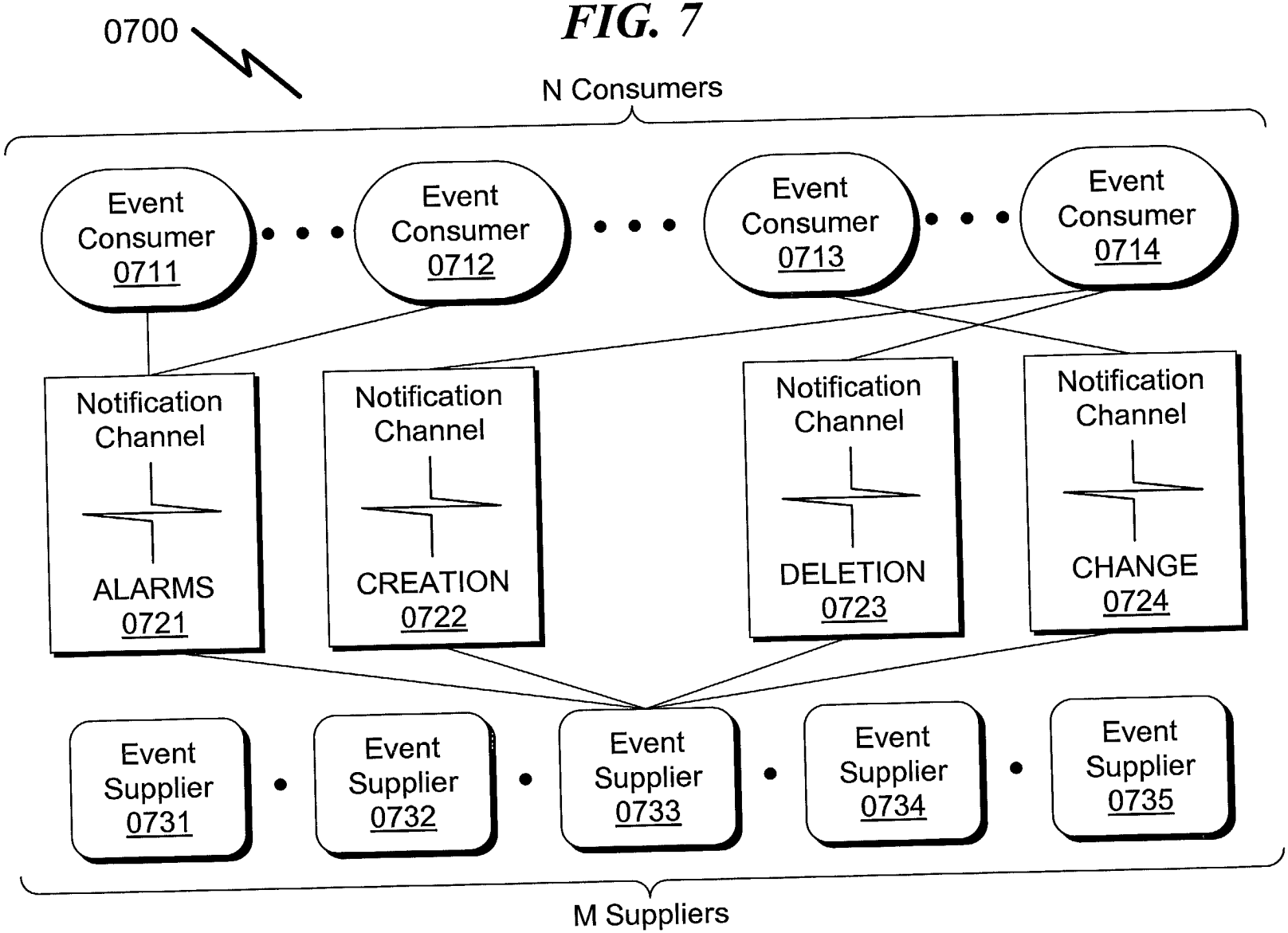
TOTTSO"9E7E5860

FIG. 6



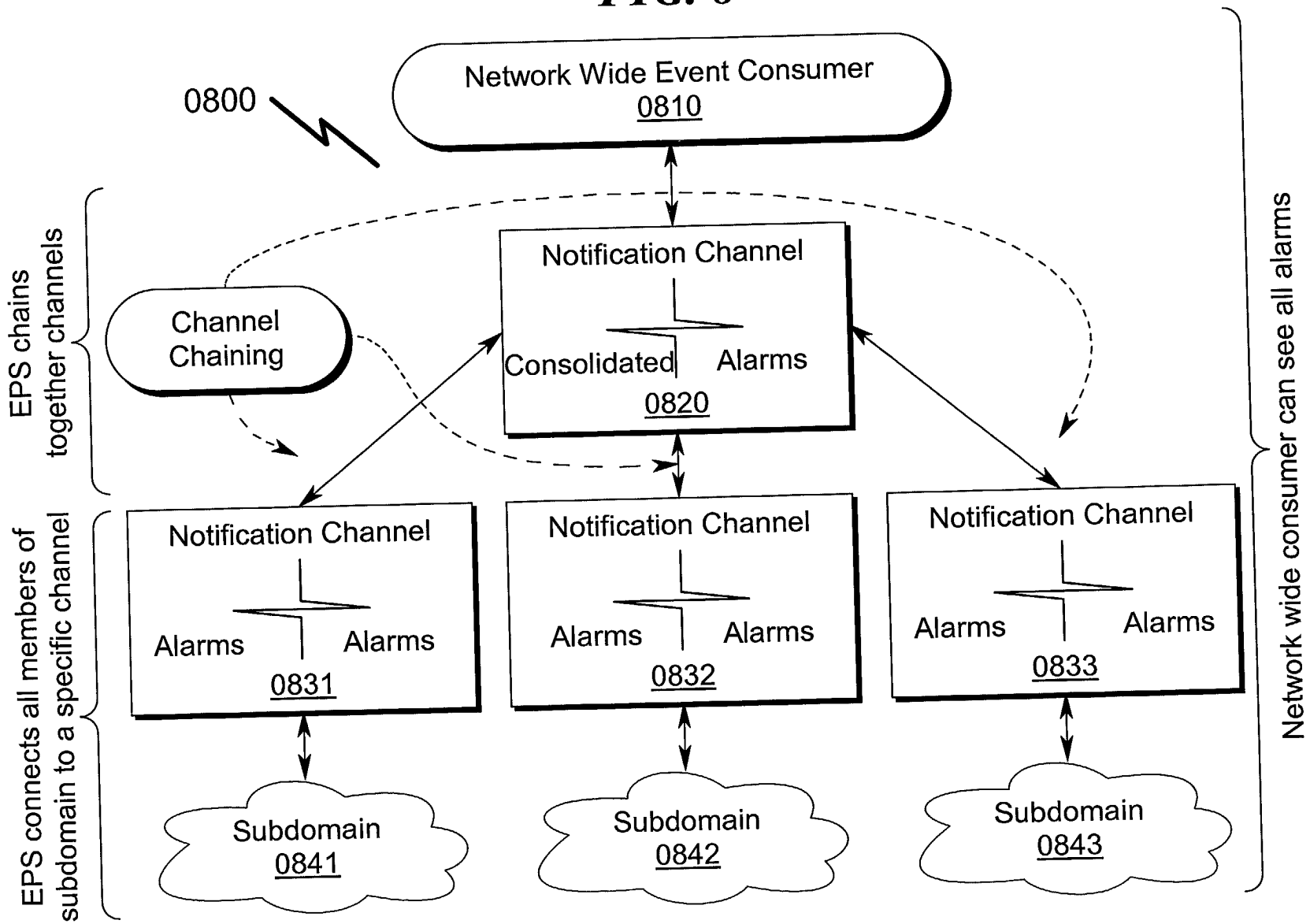
TOTTSO" 9E/ES860

FIG. 7



TOTTSO" 9E/ES860

FIG. 8



TOTTSO" 9E/ES860

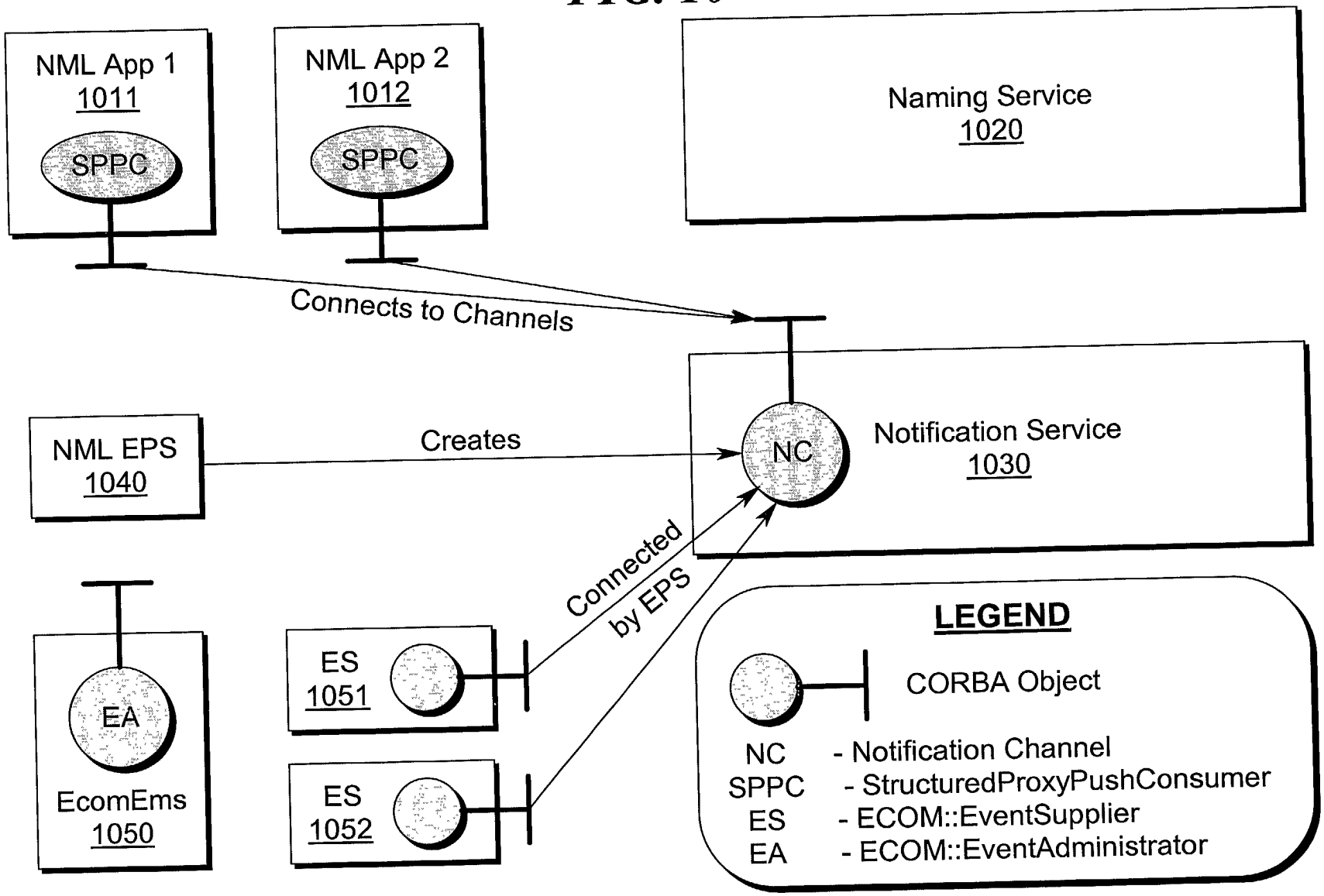
FIG. 9

0900

```
#include <CosNotification.idl>
module EventPublishingService {
    interface EventPublisher {
        EventChannel getChannel ( in PropertySeq desired_properties);
        StructuredProxyPushSupplier
        register (in  PropertySeq desired_properties,
                  in  StructuredPushConsumer a_consumer,
                  out ProxyId supplier_id);
};// end EventPublishingService
```

TOTTSO" 9E7E5860

FIG. 10



TOTTSO" 9EZ6S860

FIG. 11

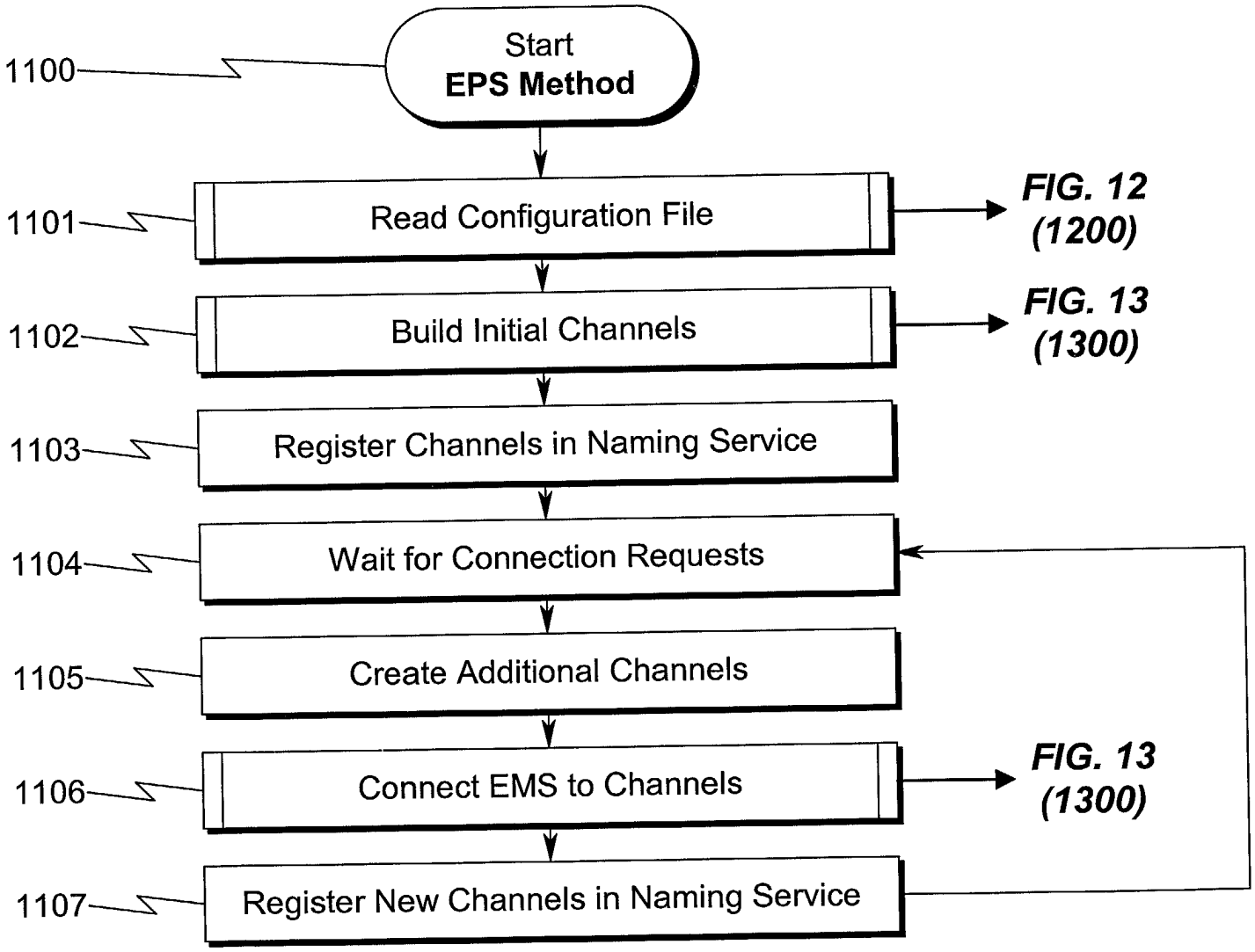


FIG. 12

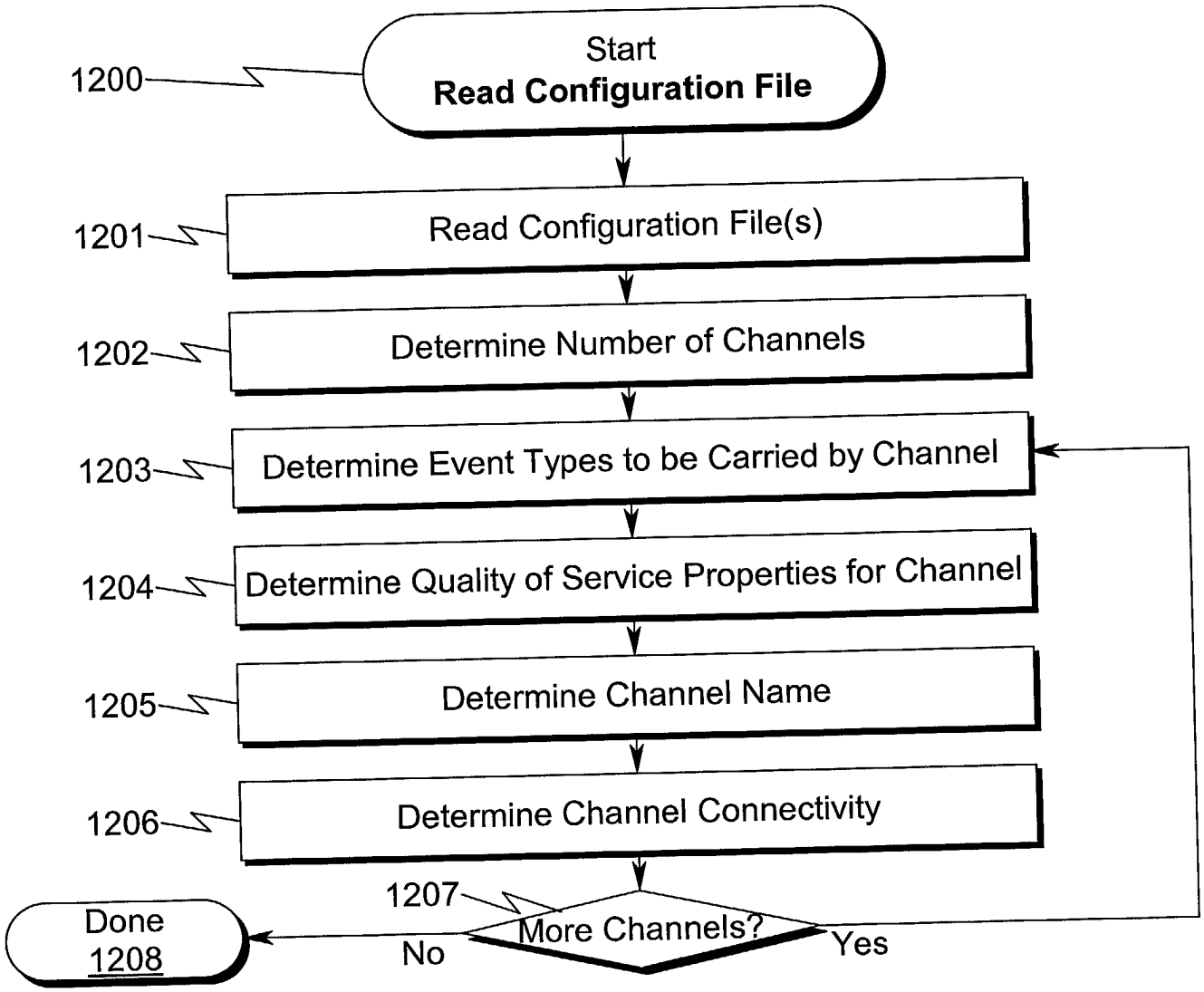

1200
Start
Read Configuration File
1201
Read Configuration File(s)
1202
Determine Number of Channels
1203
Determine Event Types to be Carried by Channel
1204
Determine Quality of Service Properties for Channel
1205
Determine Channel Name
1206
Determine Channel Connectivity
1207
More Channels?
No
Yes
Done
1208

FIG. 13

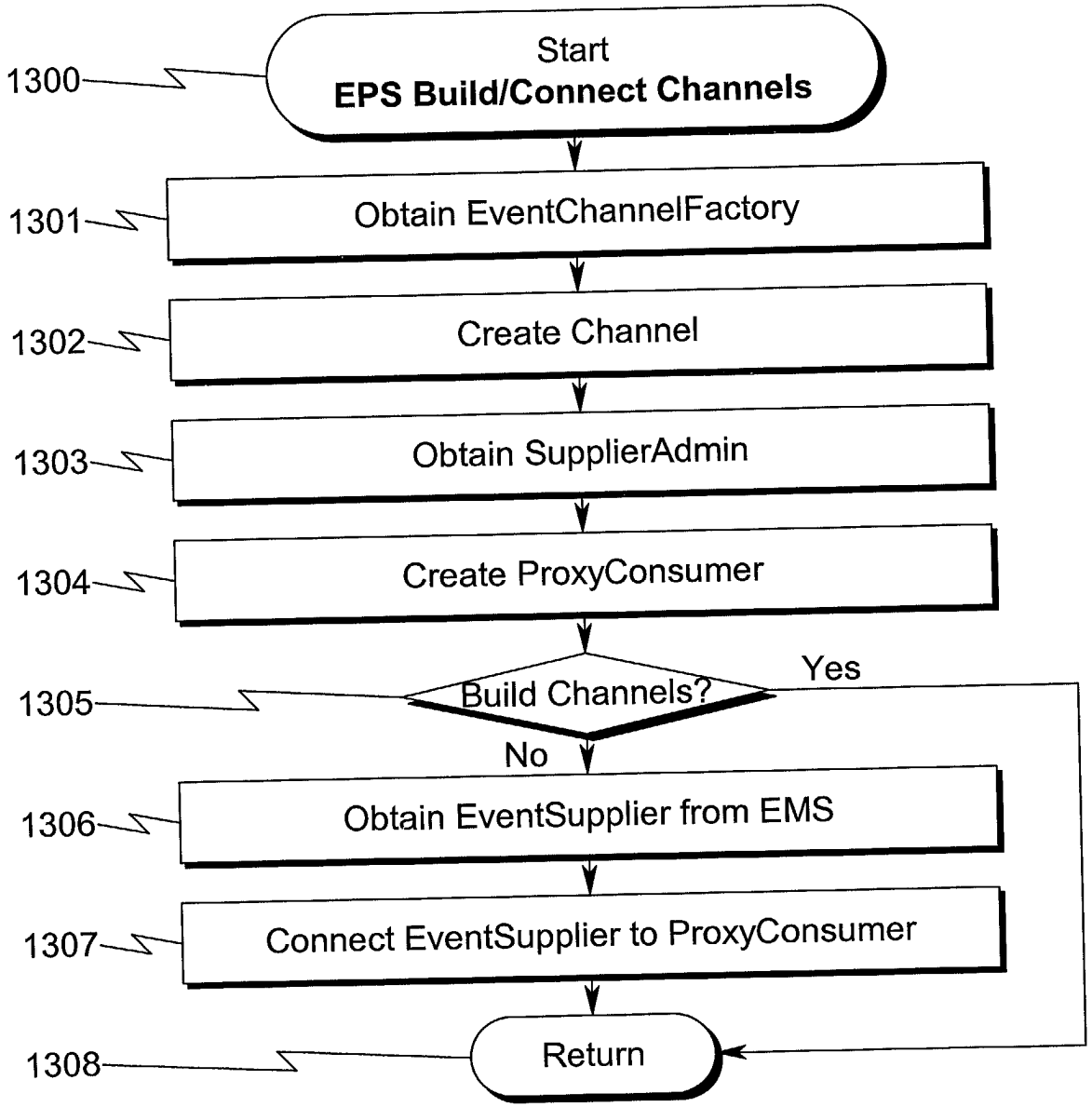

1300
Start
EPS Build/Connect Channels
1301
Obtain EventChannelFactory
1302
Create Channel
1303
Obtain SupplierAdmin
1304
Create ProxyConsumer
1305
Build Channels?
Yes
No
1306
Obtain EventSupplier from EMS
1307
Connect EventSupplier to ProxyConsumer
1308
Return

FIG. 14

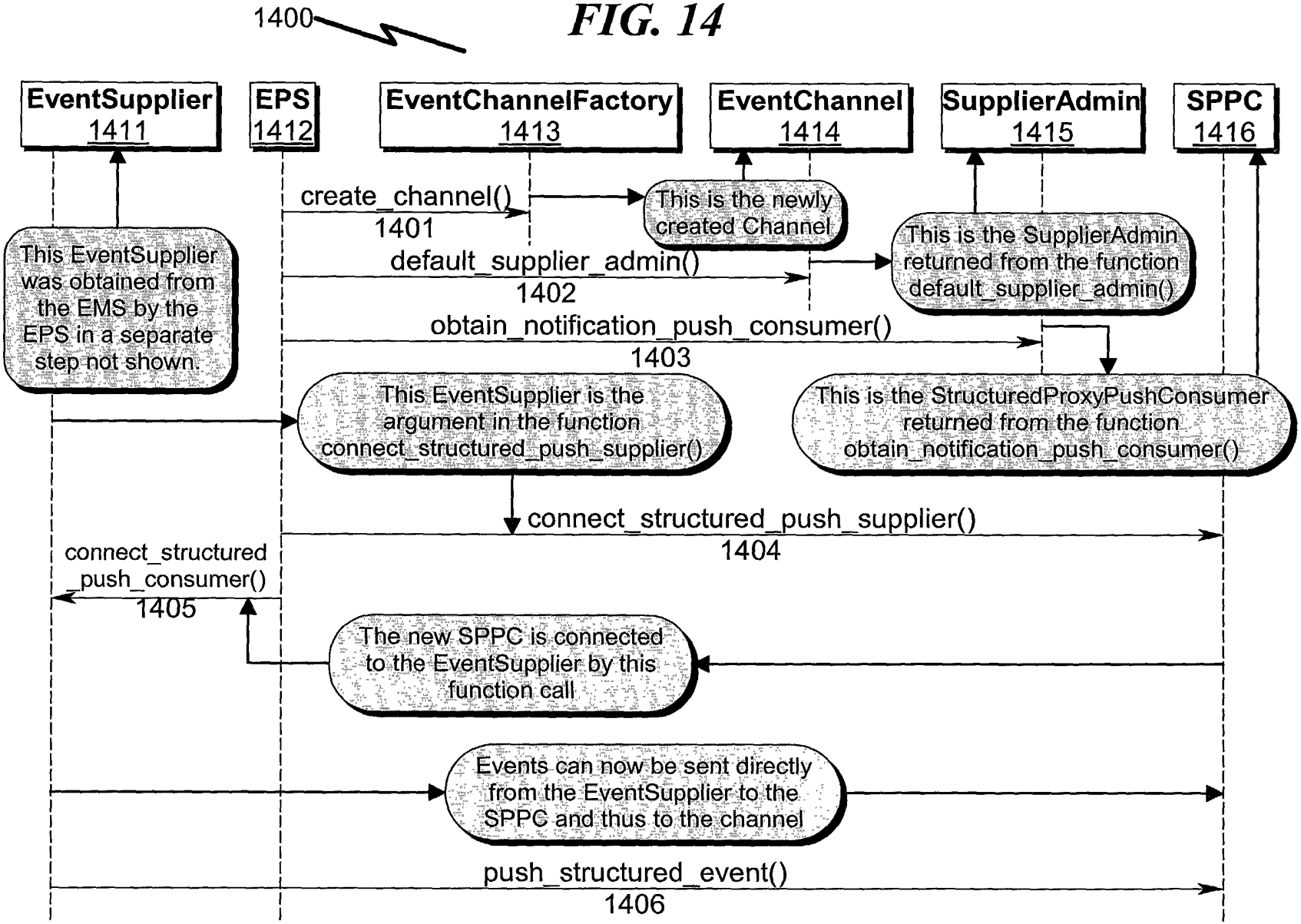

1400
EventSupplier
1411
EPS
1412
EventChannelFactory
1413
EventChannel
1414
SupplierAdmin
1415
SPPC
1416
create_channel()
1401
This is the newly created Channel
This EventSupplier was obtained from the EMS by the EPS in a separate step not shown
default_supplier_admin()
1402
This is the SupplierAdmin returned from the function default_supplier_admin()
obtain_notification_push_consumer()
1403
This EventSupplier is the argument in the function connect_structured_push_supplier()
This is the StructuredProxyPushConsumer returned from the function obtain_notification_push_consumer()
connect_structured_push_supplier()
1404
connect_structured_push_consumer()
1405
The new SPPC is connected to the EventSupplier by this function call
Events can now be sent directly from the EventSupplier to the SPPC and thus to the channel
push_structured_event()
1406

FIG. 15

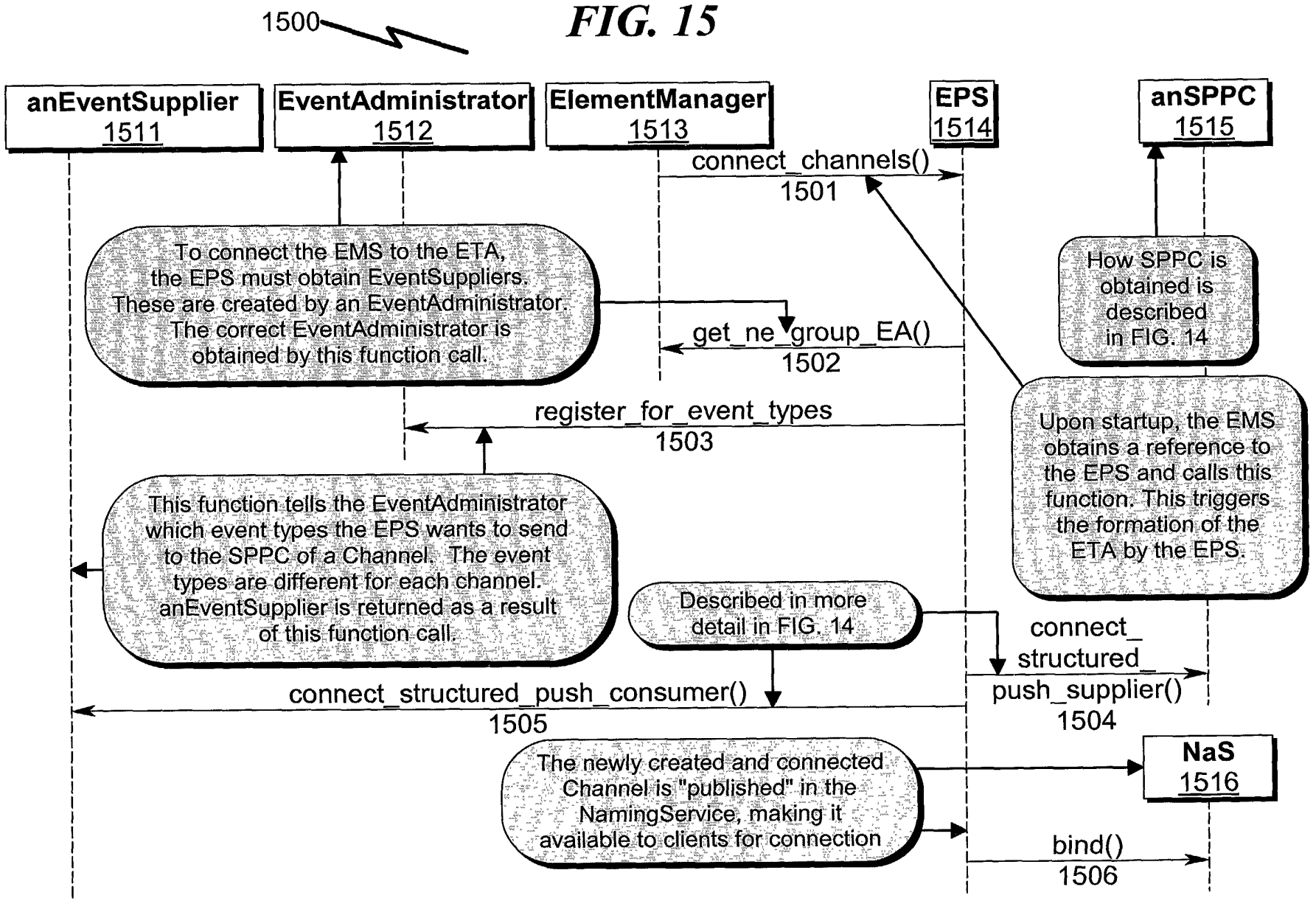
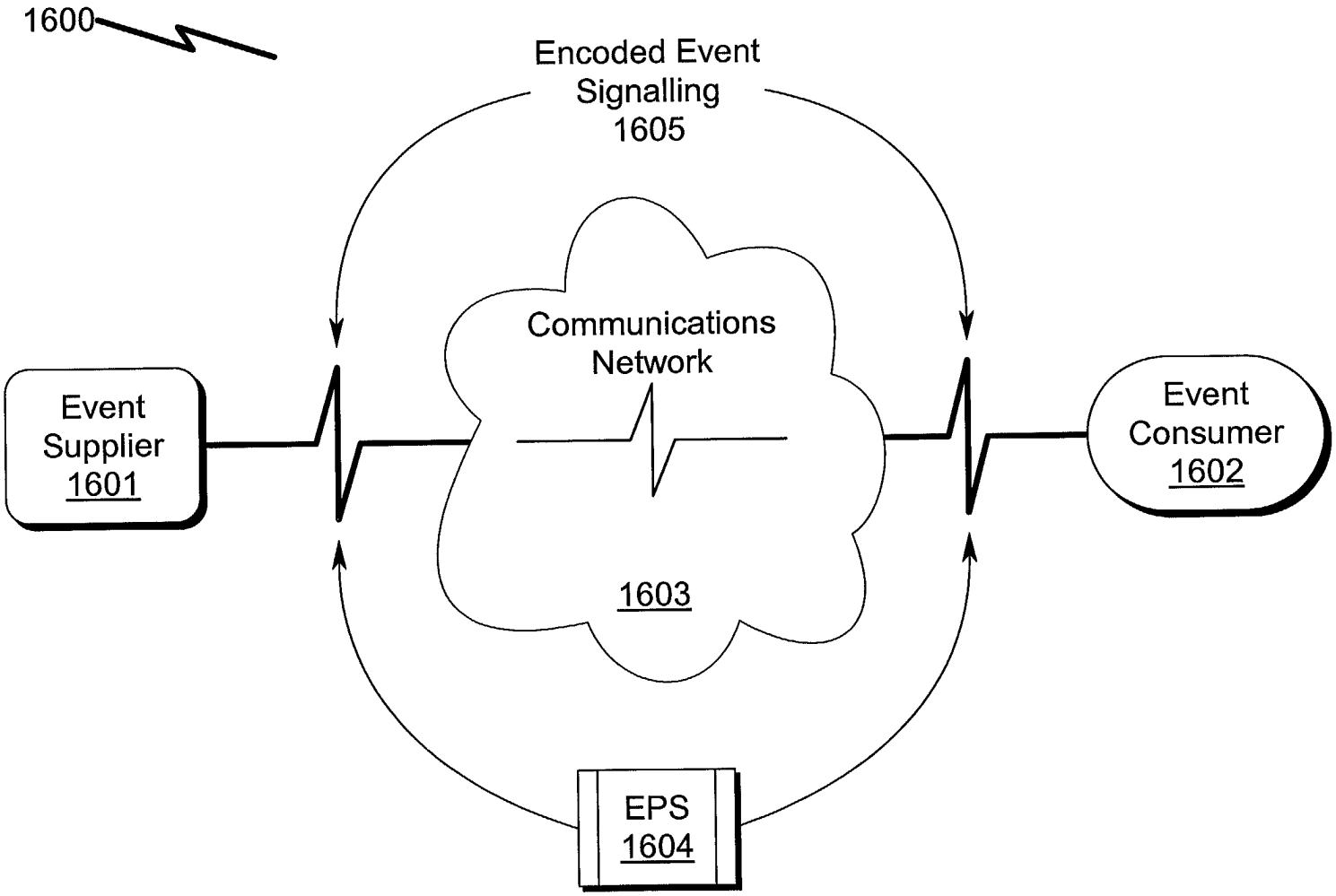

1500
anEventSupplier
1511
EventAdministrator
1512
ElementManager
1513
EPS
1514
anSPPC
1515
connect_channels()
1501
To connect the EMS to the ETA, the EPS must obtain EventSuppliers. These are created by an EventAdministrator. The correct EventAdministrator is obtained by this function call.
get_ne_group_EA()
1502
How SPPC is obtained is described in FIG. 14
register_for_event_types
1503
Upon startup, the EMS obtains a reference to the EPS and calls this function. This triggers the formation of the ETA by the EPS.
This function tells the EventAdministrator which event types the EPS wants to send to the SPPC of a Channel. The event types are different for each channel. anEventSupplier is returned as a result of this function call.
Described in more detail in FIG. 14
connect_structured_push_supplier()
1504
connect_structured_push_consumer()
1505
The newly created and connected Channel is "published" in the NamingService, making it available to clients for connection
NaS
1516
bind()
1506

FIG. 16


1600
Encoded Event Signalling 1605
Communications Network
Event Supplier 1601
Event Consumer 1602
1603
EPS 1604